Pliny Requirements Specification

1. Introduction

1.1 Purpose

This document specifies the requirements for Pliny, a web-based application for visualizing trends in Wikipedia metadata over time. It defines the functional and non-functional requirements, constraints, and success metrics for the project.

1.2 Scope

Pliny aims to collect, process, analyze, and visualize Wikipedia metadata to reveal trends in user behavior and content evolution. The system ingests metadata from various Wikipedia sources, processes it to identify significant patterns, and presents these findings through an intuitive web interface.

1.3 Definitions, Acronyms, and Abbreviations

- **Metadata**: Data about Wikipedia usage (page views, edits, etc.) rather than content itself
- **Trend**: A pattern of change observed in Wikipedia metadata over time
- **WikiMedia API**: Set of interfaces provided by the WikiMedia Foundation for accessing Wikipedia data
- **BigQuery**: Google Cloud Platform's enterprise data warehouse for analytics
- **Vandalism**: Edits made to Wikipedia pages that deliberately introduce errors or inappropriate content

2. System Overview

2.1 System Context

Pliny operates as a standalone web application that ingests data from Wikipedia's APIs and data dumps, processes this information, and presents it to users through visualizations.

2.2 User Characteristics

Target users include: - Researchers studying digital knowledge evolution - Data journalists seeking insights on public interest trends - General public interested in understanding Wikipedia usage patterns - Students and educators analyzing information consumption patterns - Wikipedia editors and administrators interested in identifying vandalism patterns

3. Functional Requirements

3.1 Data Acquisition

FR1.1: The system shall retrieve metadata from Wikipedia's APIs and data dumps. FR1.2: The system shall support ingestion of page view statistics. FR1.3: The system shall support ingestion of edit histories. FR1.4: The system shall minimize impact on Wikipedia services by optimizing query patterns. FR1.5: The system shall store retrieved data in a structured database for analysis. FR1.6: The system shall capture and track edit reversion data to identify potential vandalism.

3.2 Data Processing and Analysis

FR2.1: The system shall process raw metadata to identify notable trends. FR2.2: The system shall analyze temporal patterns in page views. FR2.3: The system shall analyze temporal patterns in edit activity. FR2.4: The system shall identify statistically significant changes in metadata patterns. FR2.5: The system shall calculate aggregate statistics across Wikipedia. FR2.6: The system shall detect and classify vandalism events based on edit patterns and reversion data. FR2.7: The system shall quantify vandalism frequency by article and track changes over time.

3.3 Visualization and User Interface

FR3.1: The system shall present processed data through interactive visualizations. FR3.2: The system shall provide a date selector to view data from specific time periods. FR3.3: The system shall display the following visualizations: - FR3.3.1: Aggregate statistics (total views, total edits) - FR3.3.2: Top viewed articles with performance trends over one week - FR3.3.3: Top edited articles as a radar chart array - FR3.3.4: Top unique editors as a bar chart - FR3.3.5: Most vandalized articles as a bar chart - FR3.3.6: Articles with most growth/reduction in bytes as a listing - FR3.3.7: Articles with most gained views, trend over last 3 days - FR3.3.8: Articles with most lost views, trend over last 3 days

FR3.4: The system shall update visualizations based on user-selected date parameters. FR3.5: The system shall present trend information in an accessible, easy-to-understand format. FR3.6: The system shall specifically highlight vandalism patterns to increase transparency about Wikipedia content manipulation.

3.4 System Architecture

FR4.1: The system shall consist of modular components: data pipeline, backend API, and frontend interface. FR4.2: The system shall enable independent operation of the data pipeline component. FR4.3: The system shall provide an API for retrieving processed trend data. FR4.4: The system shall include specialized components for vandalism detection and classification.

4. Non-Functional Requirements

4.1 Performance

NFR1.1: The data pipeline shall process daily Wikipedia metadata within 48 hours of availability. NFR1.2: The backend shall respond to API requests within 500ms under normal load. NFR1.3: The frontend shall load initial visualizations within 3 seconds on standard broadband connections.

4.2 Reliability

NFR2.1: The system shall handle interruptions in data source availability without failure. NFR2.2: The system shall maintain data integrity across processing stages. NFR2.3: The system shall provide consistent visualization results for identical time periods. NFR2.4: The system shall clearly differentiate between confirmed vandalism and potential false positives.

4.3 Usability

NFR3.1: Users shall be able to navigate between different visualizations without prior training. NFR3.2: The interface shall be accessible on desktop devices. NFR3.3: Visualization interactions shall follow standard web conventions for usability. NFR3.4: Vandalism visualizations shall be easily understandable to non-technical users.

4.4 Security

NFR4.1: The system shall only use publicly available Wikipedia data. NFR4.2: The system shall not collect personally identifiable information from users. NFR4.3: The system shall avoid exposing personally identifiable information of Wikipedia editors.

4.5 Maintainability

NFR5.1: The system shall use modular architecture to facilitate updates to individual components. NFR5.2: The system shall include documentation for each component. NFR5.3: The system code shall follow language-specific best practices.

4.6 Scalability

NFR6.1: The data pipeline shall support processing increasing volumes of Wikipedia metadata. NFR6.2: The database shall accommodate at least 3 years of historical metadata.

5. System Constraints

5.1 Technical Constraints

- The data pipeline must be implemented in Python 3.11+
- The backend must be implemented in Go 1.23+
- The frontend must be implemented using React/TypeScript
- The system must use BigQuery for data storage

5.2 Business Constraints

- The system must operate within reasonable cloud infrastructure costs
- The system must comply with Wikipedia's terms of service and rate limits
- The system must present vandalism data responsibly without encouraging disruptive behavior

6. Success Criteria

6.1 Technical Success Criteria

- Successfully ingest and process Wikipedia metadata dating back to September 2023
- Generate meaningful trend visualizations from processed data
- Deploy a publicly accessible web application
- Create a reusable data pipeline component for future projects
- Effectively identify and visualize vandalism patterns across Wikipedia

7. Future Considerations

- Support for non-English Wikipedia metadata
- Extension of historical data coverage beyond September 2023
- Analysis of trends over longer time periods (months/years)
- Additional visualization types and interaction patterns